PROCEDURE OF BUILDING STOCK MARKET APPLICATION IN GCP CLOUD:

In this project we are going to run or deploy a stock market app like (ex:grow,zerodha ect) in google cloud.

Services used in this project:

* Cloud run.
* Cloud sql.
* Cloud storage bucket.
* Cloud scheduler.
* Cloud security.

Project overview:

This app is basically a software that has running between Monday to Friday and should have zero downtime between 9-15am to 3-45 pm between mentioned days so we choose cloud run for deploying this application and used the cloud sql to schedule and order purpose of our stocks, and used the storage bucket to handle all the files related to the customer and customer information, we used cloud scheduler and cloud pub/sub to trigger the price of the stock for every 5minutes. Each and every project in GCP or outside requires security and some authority that who should use the software and who have the rights to edit or use to have rights to make changes in the running application and where should it run (ex:zone or region) and when and why to change the application to requirements so for that purpose we used cloud IAM and cloud security.

Below are the some of the screen shots of the running project and services:

1. **The first step in every GCP project is to create a new project in google cloud console.**

* **Navigate to Google Cloud Console:**
  + Open the [Google Cloud Console](https://console.cloud.google.com/).
* **Select or Create a Project:**
  + In the top right corner, click on the project dropdown (next to the organization name).
  + Click on "New Project."
* **Fill in Project Details:**
  + Enter a name for your project.
  + Optionally, you can edit the project ID (it should be unique).
  + Choose an organization (if applicable).
  + Select a billing account.
* **Configure Additional Settings (Optional):**
  + You can configure advanced settings like adding the project to a folder or enabling the Cloud Resource Manager API.
* **Review and Create:**
  + Review the project details.
  + Click on the "Create" button.
* **Wait for Project Creation:**
  + It may take a few moments for the project to be created. Once completed, you will see a notification.
* **Access the New Project:**
  + After the project is created, you can access it from the project dropdown in the top right corner.

A screenshot of a computer

Description automatically generated

Pic 1.1This above picture shows the new project console

A screenshot of a computer

Description automatically generated

**2.clone the source repository,creating a docker image and container with cloud run, deploy the container or a docker image to cloud run:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

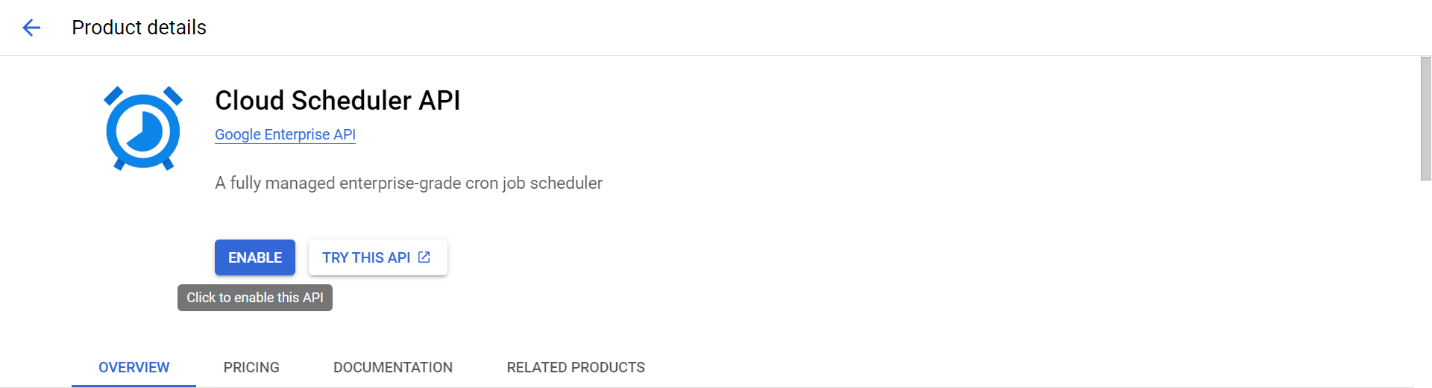
Procedure:

1. **Clone the Source Repository:**
   * Use Git to make a local copy of the project from a version control repository.
2. **Navigate to the Project Directory:**
   * Move into the cloned project directory using the command line.
3. **Create a Docker Image:**
   * Build a Docker image using a Dockerfile, specifying dependencies and configurations for your application.
4. **Push the Docker Image to Google Container Registry:**
   * Authenticate Docker with Google Container Registry and upload your Docker image to the registry.
5. **Deploy the Container to Cloud Run:**
   * Deploy the Docker image to Cloud Run, a fully managed serverless container platform on Google Cloud.
6. **Access the Deployed App:**
   * Once deployed, access your application using the provided URL.

**3.Creation of service account for the services:**

1. **Create Service Account for Cloud Run:**
   * Navigate to the [IAM & Admin section](https://console.cloud.google.com/iam-admin) in the Google Cloud Console.
   * Click on "Service accounts" in the left sidebar.
   * Click on "Create Service Account," provide a name like **cloud-run-service-account**, and assign roles such as **roles/run.admin** for Cloud Run management.
2. **Create Service Account for Cloud SQL:**
   * In the same "Service accounts" section:
   * Click on "Create Service Account," provide a name like **cloud-sql-service-account**, and assign roles such as **roles/cloudsql.admin** and **roles/cloudsql.editor** for Cloud SQL management.
3. **Create Service Account for Cloud Storage:**
   * Still in the "Service accounts" section:
   * Click on "Create Service Account," provide a name like **cloud-storage-service-account**, and assign roles such as **roles/storage.admin** and **roles/storage.objectAdmin** for Cloud Storage management.
4. **Create Service Account for Cloud Scheduler and Cloud Pub/Sub:**
   * Continue in the "Service accounts" section:
   * Click on "Create Service Account," provide a name like **cloud-scheduler-service-account**, and assign roles such as **roles/cloudscheduler.admin** and **roles/pubsub.admin** for Cloud Scheduler and Cloud Pub/Sub management.
5. **Create IAM Role for App Administrators:**
   * Navigate to the [IAM & Admin section](https://console.cloud.google.com/iam-admin) in the Google Cloud Console.
   * Click on "Roles" in the left sidebar.
   * Click on "Create Role," provide a name like **app-admin**, and define the necessary permissions for managing Cloud Run, Cloud SQL, Cloud Storage, Cloud Scheduler, and Cloud Pub/Sub.
6. **Create IAM Role for Application Users:**
   * In the same "Roles" section:
   * Click on "Create Role," provide a name like **app-user**, and define the necessary permissions for users interacting with the application.
7. **Create IAM Role for Security and Compliance:**
   * Similarly, create a role named **security-compliance** with permissions related to monitoring and ensuring compliance with security policies.
8. **Create IAM Role for Change Management:**
   * Create a role named **change-management** with permissions for deploying updates to Cloud Run, modifying database structures in Cloud SQL, and managing file storage in Cloud Storage.
9. **Create IAM Role for Regional and Zonal Authorities:**
   * Create a role named **regional-zonal-authority** with permissions for managing resources in specific geographic locations.
10. **Create IAM Role for Incident Response:**
    * Create a role named **incident-response** with permissions for troubleshooting issues, accessing logs, and taking corrective actions.

**4.Creating cloud pub/sub and cloud scheduler for triggering the task**



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Procedure:

1. **Cloud Scheduler:**
   * **Enable API:**
     + Ensure that the Cloud Scheduler API is enabled for your Google Cloud project.
   * **Create a Pub/Sub Topic:**
     + Create a Pub/Sub topic in the Google Cloud Console. This topic will be used to send messages when the scheduled task is triggered.
   * **Create a Job:**
     + Go to the Cloud Scheduler section in the Google Cloud Console.
     + Click on "Create Job."
     + Provide a name for the job, set the frequency to every 5 minutes (using the cron syntax **\*/5 \* \* \* \***), and choose the appropriate time zone.
   * **Configure Target:**
     + Choose "Pub/Sub" as the target.
     + Provide the Pub/Sub topic name you created earlier.
   * **Configure Payload:**
     + If your Pub/Sub message requires a payload, you can provide it in the "Payload" section.
   * **Configure Authentication:**
     + If your Pub/Sub topic requires authentication, configure the authentication details in the "Authentication" section.
   * **Create the Job:**
     + Click on "Create" to create the Cloud Scheduler job.
2. **Pub/Sub:**
   * **Subscription:**
     + Create a subscription for the Pub/Sub topic to which Cloud Scheduler sends messages.
   * **Pub/Sub Message Handler:**
     + Implement a message handler that processes the stock price update task when a message is received on the Pub/Sub subscription.
   * **Error Handling (Optional):**
     + Implement error handling mechanisms in your Pub/Sub message handler to deal with potential issues during the processing of the stock price update task.
   * **Monitoring and Logging (Optional):**
     + Set up monitoring and logging for your Pub/Sub topic and subscription to track the execution of the scheduled task and troubleshoot any issues.
3. **Update Stock Prices Task:**
   * Implement the logic for updating stock prices in the message handler of your Pub/Sub subscription. This logic will be executed every time Cloud Scheduler triggers the scheduled task.
4. **Deploy and Monitor:**
   * Deploy your updated code to the appropriate environment.
   * Monitor the Cloud Scheduler job execution and the logs generated by your Pub/Sub message handler to ensure that the stock prices are updated as expected.

**5.Use of cloud sql for ordering and history of stock:**

1. **Set Up Cloud SQL Instance:**
   * Create a Cloud SQL instance in the GCP Console with necessary configurations.
2. **Create Database and Tables:**
   * Design a database with tables for stocks, orders, and stock history.
3. **Cloud SQL Connection in App:**
   * Integrate Cloud SQL connection details into your app for database access.
4. **Ordering a Stock:**
   * Implement logic to record stock orders in the **orders** table.
5. **Update Stock Quantity:**
   * Adjust stock quantities in the **stocks** table based on order execution.
6. **Save Stock History:**
   * Store transaction details in the **stock\_history** table.
7. **Querying Stock Information:**
   * Develop functionality to retrieve stock details and transaction history.
8. **Error Handling and Logging:**
   * Implement error handling and logging for smooth operations and debugging.
9. **Security Considerations:**
   * Ensure secure Cloud SQL connections and prevent SQL injection.
10. **Testing and Monitoring:**
    * Thoroughly test the ordering process and set up monitoring for database interactions.

**6.Creating Cloud Storage:**

1. **Navigate to Cloud Storage in GCP Console:**
   * Open the [Google Cloud Console](https://console.cloud.google.com/).
   * In the left navigation pane, select "Storage" > "Cloud Storage."
2. **Create a Bucket:**
   * Click on the "Create Bucket" button.
   * Provide a globally unique name for your bucket.
   * Choose a default storage class and location for your data.
   * Set access control options if needed (e.g., fine-grained permissions).
3. **Configure Advanced Settings (Optional):**
   * Adjust additional settings such as versioning, object lifecycle management, and logging based on your project requirements.
4. **Create the Bucket:**
   * Click on "Create" to create the Cloud Storage bucket.

**Use of Cloud Storage in the Stock Market App:**

1. **File Storage for Customer Data:**
   * Upload and store customer-related files, such as documents or images, in the Cloud Storage bucket associated with each customer profile.
2. **Historical Stock Data:**
   * Archive historical stock data, reports, or other documents in Cloud Storage, providing a scalable and durable solution for storing large datasets.
3. **Static Asset Hosting:**
   * Store static assets like images, stylesheets, or client-side scripts in Cloud Storage. This allows for efficient hosting and delivery of these assets to users.
4. **Backup and Recovery:**
   * Perform regular backups of critical app data to Cloud Storage, ensuring data integrity and providing a mechanism for recovery in case of data loss.
5. **Integration with Cloud Run:**
   * Cloud Run services can easily access and retrieve data stored in Cloud Storage, facilitating seamless integration between serverless components of the app.
6. **Secure Storage for Application Assets:**
   * Ensure that sensitive assets, such as configuration files or application-specific resources, are securely stored in Cloud Storage with appropriate access controls.
7. **Data Sharing and Collaboration:**
   * Share relevant data or documents among different components of the app or collaborate with external stakeholders by providing controlled access to specific Cloud Storage objects.
8. **Cost-Effective Storage Solution:**
   * Leverage Cloud Storage's cost-effective pricing model, paying only for the storage capacity and data transfer you use.
9. **Scalability and Redundancy:**
   * Benefit from Cloud Storage's automatic scalability and redundancy, ensuring high availability and performance even as data volumes grow.
10. **Object Lifecycle Management:**
    * Use object lifecycle management to define rules for automatically managing objects over time, such as transitioning data to cheaper storage classes or deleting obsolete data.

**7.cloud monitoring and logging:**

**Logging Services:**

1. Enable Cloud Logging API:
   * In the GCP Console, go to "APIs & Services" > "Dashboard."
   * Search for "Cloud Logging API" and enable it.
2. View Logs:
   * Navigate to "Logging" in the GCP Console.
   * Here, you can view logs, filter by severity, and search for specific log entries.
3. Custom Logs:
   * Integrate logging into your application code using Cloud Logging libraries. Customize log statements to capture specific events.
4. Log Sinks:
   * In the Logging section, go to "Exports."
   * Set up log sinks to export logs to Cloud Storage or BigQuery for long-term storage and analysis.

Monitoring Services:

1. Enable Cloud Monitoring API:
   * In the GCP Console, go to "APIs & Services" > "Dashboard."
   * Search for "Cloud Monitoring API" and enable it.
2. Create Dashboards:
   * Go to "Monitoring" in the GCP Console.
   * Create dashboards to visualize key metrics and performance indicators. Click "Create Dashboard" and add relevant charts.
3. Create Alerts:
   * In the Monitoring section, go to "Alerting" > "Create Policy."
   * Define conditions and thresholds for metrics. Set up notifications to receive alerts when conditions are met.
4. Uptime Checks:
   * In the Monitoring section, go to "Uptime Checks" > "Add Uptime Check."
   * Configure checks to monitor the availability of your stock market app at regular intervals.
5. Incident Management:
   * In the Monitoring section, go to "Alerting" > "Incident Management."
   * Set up incident management policies to streamline responses to issues, integrating with communication channels.
6. Custom Metrics:
   * In the Monitoring section, go to "Metrics Explorer."
   * Create custom metrics to track application-specific performance indicators.

Integration with Other Services:

1. Logging to Cloud Storage:
   * In the Logging section, go to "Exports."
   * Set up an export to Cloud Storage, specifying the bucket and object prefix.
2. Pub/Sub for Logging Events:
   * In the Logging section, go to "Exports."
   * Set up a Pub/Sub export for logs, enabling event-triggered actions based on log entries.
3. Enable Cloud Error Reporting:
   * In the Error Reporting section, go to "Error Reporting" > "Enable API."
4. Enable Cloud Trace and Profiler:
   * In the Monitoring section, go to "Tracing" > "Enable Tracing."
   * In the Profiler section, go to "Enable Profiler" and follow the instructions.